Deep Just-In-Time Inconsistency Detection Between Comments and Source Code

Sheena Panthaplackel (spantha@cs.utexas.edu), Junyi Jessy Li, Milos Gligoric, Raymond J. Mooney
The University of Texas at Austin

The University of Texas at Austin

Problem

When developers make code changes, they often fail to update comments accordingly. This results in inconsistent comments that lead to time-wasting confusion and vulnerability to bugs.

✘

```
/**Computes the highest value from the list of scores.*/
public int getBestScore() {
-       return Collections.max(scores);
+       return Collections.min(scores);
}
```

Goal: Determine whether a comment is inconsistent, just-in-time, i.e. right before code changes are merged into a code base.

Approach

Data: Mining comment-method pairs from consecutive commits (C, M_{old}), (C_{new}, M) with code changes (i.e., $M_{old} \neq M$), from open-source Java projects on GitHub.

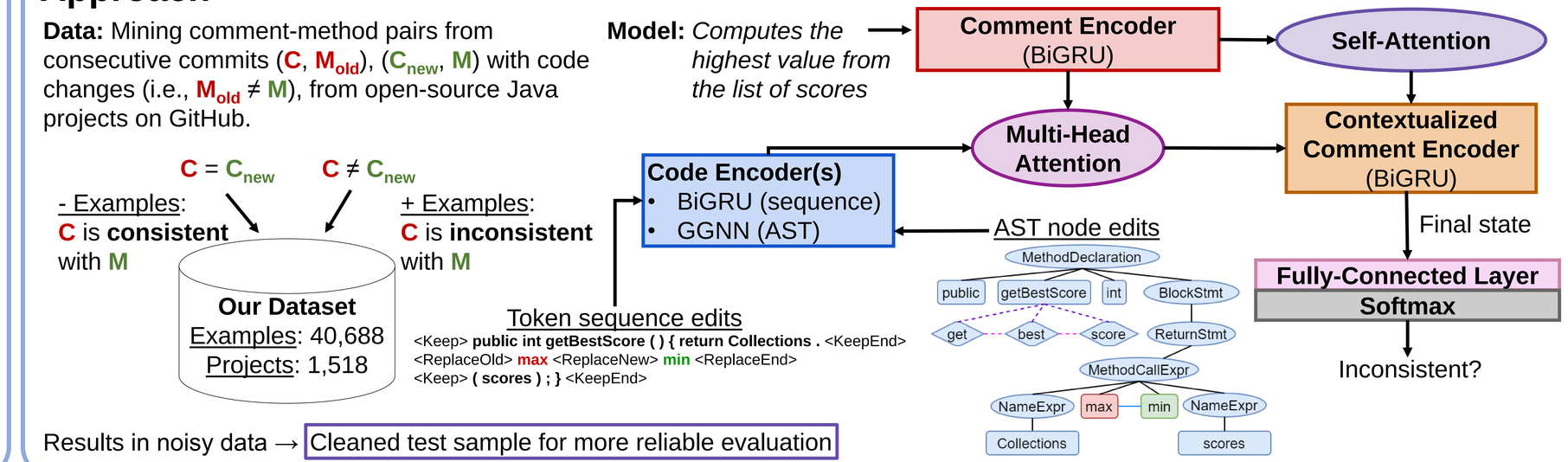
$C = C_{new}$ → - Examples: **C** is **consistent** with **M**

$C \neq C_{new}$ → + Examples: **C** is **inconsistent** with **M**

Our Dataset
Examples: 40,688
Projects: 1,518

Results in noisy data → Cleaned test sample for more reliable evaluation

Model: *Computes the highest value from the list of scores*



Evaluation

Intrinsic Evaluation

	F1	Acc
Liu et al. (2018)	75.8	76.3
Post Hoc SEQ	63.0	60.3
Just-In-Time SEQ	81.5	82.0
Just-In-Time GRAPH	81.4	82.0
Just-In-Time HYBRID	83.1	83.8
Just-In-Time HYBRID + features	**87.1**	**87.8**

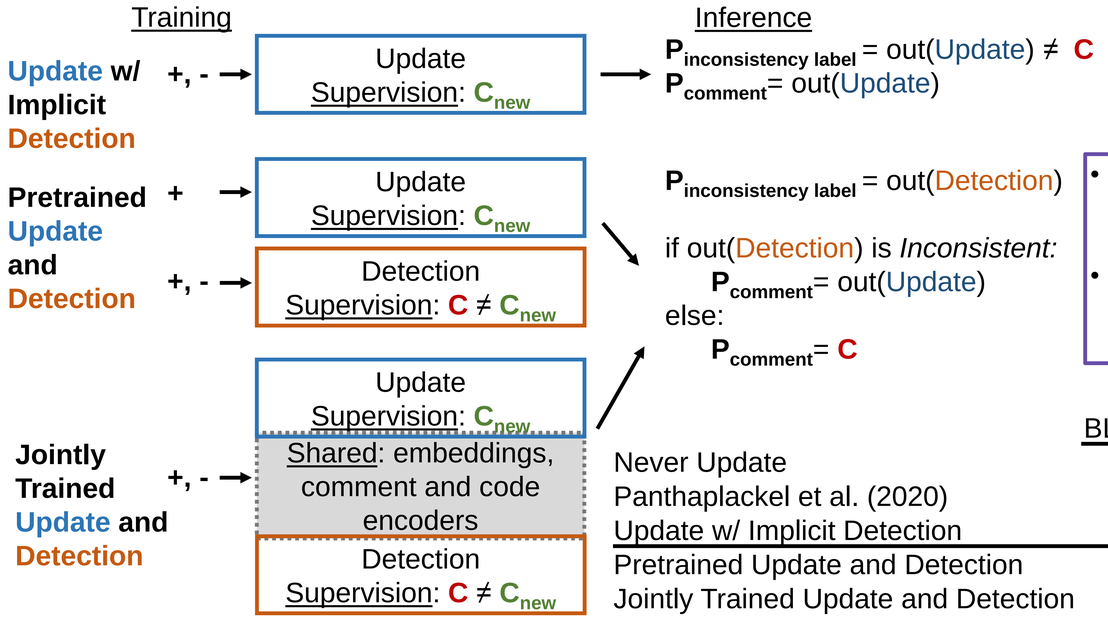
No significant difference between SEQ, GRAPH, and HYBRID models.

Incorporating auxiliary features can further boost performance.

Just-In-Time approaches outperform post hoc and baseline models.

Extrinsic Evaluation: Evaluating a comprehensive comment maintenance system which automatically updates a comment *(Panthaplackel et al., ACL 2020)* if inconsistency is detected by our newly proposed approach.

```
/**Computes the lowest ~~highest~~ value from the list of scores.*/
```



- Explicit inconsistency detection leads to improved performance
- Pretrained and jointly trained systems perform similarly

	BLEU-4	GLEU	F1	Acc
Never Update	72.1	68.2	0.0	50.0
Panthaplackel et al. (2020)	68.7	67.4	69.0	57.1
Update w/ Implicit Detection	74.7	72.7	37.7	61.7
Pretrained Update and Detection	**77.0**	**76.2**	86.4	87.1
Jointly Trained Update and Detection	76.6	75.6	**87.2**	**87.3**